

# Dokumentation Studienarbeit E-Mail-Programm

im Fach

.NET Programmierung mit C#

von

Daniel Glaser

im Studiengang

Mobile Computing

an der Hochschule für angewandte Wissenschaften Hof

## Inhaltsverzeichnis

---

1	Einführung .....	3
2	Beschreibung der Aufgabenstellung.....	4
2.1	Vorgegebene Anforderungen .....	4
2.2	Eigene Anforderungen .....	4
3	Implementierung: Beschreibung der Lösung .....	5
3.1	Angaben zum Design der Oberfläche .....	5
3.2	Angaben zur Umsetzung .....	5
3.2.1	Beschreibung der Komponenten .....	5
3.2.2	Verwendete Algorithmen, Bibliotheken oder Techniken .....	6
3.2.3	Verwendeter fremder Quelltext .....	6
4	Benutzerhandbuch .....	8
4.1	Hauptfenster .....	8
4.2	Optionen .....	8
4.2.1	Account anlegen .....	8
4.2.2	Account bearbeiten .....	8
4.2.3	Account als Standard definieren.....	9
4.2.4	Account löschen.....	9
4.3	E-Mail schreiben .....	9
4.4	E-Mails abholen .....	9
4.5	E-Mails anzeigen .....	9
4.5.1	E-Mail beantworten.....	9
4.5.2	E-Mail weiterleiten .....	9
4.5.3	E-Mail als ungelesen markieren.....	9
5	Diskussion .....	10
5.1	Zusammenfassung der Ergebnisse.....	10
5.2	Aufgetretene Probleme .....	10
5.2.1	WebBrowser ruckelt .....	10
5.2.2	Testen des SMTP-Server .....	10
5.2.3	Speichern von Anhängen .....	10
5.2.4	InvalidOperationException in der doEmailsContainsId .....	10
5.2.5	E-Mails abholen im gemeinsamen Posteingang .....	10
5.3	Weiterentwicklung des Programmes.....	10
6	Eidesstattliche Erklärung .....	11

## 1 Einführung

---

Ein E-Mail-Programm gehört zu den grundlegendsten Programmen, die es gibt, weil es so gut wieder jeder braucht. Viele E-Mail-Programme sind aber mit Funktionen überladen und brauchen deswegen lange zum Starten oder haben eine unübersichtliche Oberfläche. Deswegen habe ich mir in dieser Studienarbeit als Ziel genommen, ein E-Mail-Programm zu programmieren, welches schnell startet und eine übersichtliche Oberfläche hat, damit es leichter zu bedienen ist. Es soll flüssig laufen und die wichtigsten Funktionen bieten.

Als Vorbild habe ich mir andere E-Mail-Programme wie z. B. Thunderbird aber auch Online-E-Mail-Dienste wie z. B. Inbox herangezogen.

## 2 Beschreibung der Aufgabenstellung

---

### 2.1 Vorgegebene Anforderungen

Als Studienarbeit wird ein mit C# und XAML erstelltes Programm mit grafischer Oberfläche entwickelt, dass folgende Anforderungen erfüllt:

- Programm verwendet bzw. verarbeitet Daten, die aus einer beliebigen Quelle (z.B. Webservice, Datenbank, Datei) stammen können
- Daten müssen gespeichert und geladen werden können
- Nutzung von WPF für die GUI
  - Ansprechendes Design der GUI
  - Verwendung passender Steuerelemente
  - Layout der GUI passt sich an Größenänderungen des Fensters an
- Oberfläche wird in XAML-Dateien definiert, nicht im Code erzeugt
- Konsequente Nutzung des MVVM-Patterns, beachten Sie dazu:
  - Aufteilung in Model, View und ViewModel
  - Datenbindung
  - Code-Behind enthält nur den nötigsten Code
- Trennung in 2 Teilprojekte:
  - Common (Class Library (Portable) oder Class Library)
    - Enthält Model-Klassen, Klassen für Datenzugriff sowie sonstige Klassen (z.B. für Utilities)
    - Enthält ViewModels
    - Targets für Portable Class Library (mind.): sh. Abbildung 1
  - WpfView (WPF Application)
    - Referenz auf Projekt „Common“
    - Enthält nur die Views sowie direkt dazugehörigen Code
    - Target Framework: 4.5 bzw. 4.6 (passend zum anderen Projekt!)
- Aufteilung der Quellcodedateien in eine sinnvolle Klassenhierarchie, d.h. Verwendung von Namespaces. Ein einfaches, nicht vollständiges Beispiel sh. Abbildung 2.
- Objektorientierte Programmierung
- Anwendung asynchroner Programmierung bei länger dauernden Aufgaben
- Mindestens 1500 Lines of Code

### 2.2 Eigene Anforderungen

Es soll ein E-Mail-Programm entstehen, das zuerst einmal die Grundfunktionen E-Mail schreiben und E-Mails lesen ermöglichen soll. Dabei sollen HTML E-Mails richtig dargestellt werden.

Es soll möglich sein zu erkennen, welche E-Mails bereits gelesen wurden und auch eine E-Mail wieder als ungelesen markieren zu können.

Die E-Mails sollen auch gelöscht werden können. Lokal aber auch auf dem Server.

Eine E-Mail soll man weiterleiten können und darauf antworten können. Ebenfalls soll es möglich sein allen zu antworten an die die E-Mail gesendet wurde.

Bei einer E-Mail soll es möglich sein einen Anhang mit zu verschicken. Und bei E-Mails mit Anhang soll der Anhang zu öffnen oder speichern sein.

Es soll die Möglichkeit geben mehrere Accounts anzulegen und zu verwalten.

## 3 Implementierung: Beschreibung der Lösung

---

### 3.1 Angaben zum Design der Oberfläche

Beim Design wurde sich an Thunderbird orientiert aber trotzdem ein Eigenes erstellt.

Als Farben wurden, weiß und blau verwendet, da es zu dem modernen Flat Design passen soll.

### 3.2 Angaben zur Umsetzung

#### 3.2.1 Beschreibung der Komponenten

##### 3.2.1.1 Model

Email

Die Email-Klasse ist eine selbst erstellte Klasse die eine E-Mail repräsentiert. Sie beinhaltet alles was für dieses E-Mail-Programm von einer E-Mail benötigt wird.

Account

Diese Klasse repräsentiert einen E-Mail-Account mit allen Parametern, die ein E-Mail-Account in diesem Programm benötigt.

Es beinhaltet eine Liste aller E-Mails, die zu diesem Account gehören.

##### 3.2.1.2 Services

EmailService

Der EmailService kümmert sich darum E-Mails zu senden und zu empfangen. Außerdem hat er noch Methoden, mit denen der IMAP, POP3 und SMTP Server getestet werden können.

E-Mails senden

Zum Senden von E-Mails wird der SmtpClient von .NET verwendet.

E-Mails abholen

Für das Abholen von E-Mails wird das MailKit verwendet. Entweder wird der ImapClient oder der Pop3Client verwendet, je nachdem welche Verbindung der Nutzer eingestellt hat.

Server testen

Zum Testen der Server werden die 3 Clients des MailKits verwendet, um sich mit dem jeweiligen Server zu verbinden. Dabei wird überprüft, ob es erfolgreich war oder nicht und ein true zurückgegeben oder Exceptions geworfen.

DataService

Der DataService kümmert sich um das Speichern und Laden der Daten. Die Daten sind einmal die Accounts mit den E-Mails. Und dann noch den Index des als Standards festgelegten Accounts.

##### 3.2.1.3 ViewModel

EmailViewModel

Das EmailViewModel beinhaltet alles, was auch eine E-Mail beinhaltet aber die Listen sind hier ObservableCollection.

AccountViewModel

Das AccountViewModel beinhaltet alles, was auch ein Account beinhaltet aber die Listen sind hier ObservableCollection.

AccountListViewModel

Das AccountListViewModel beinhaltet eine Liste aller AccountViewModels. Sie hat zudem Methoden, mit denen sie auf die MailService Klasse zugreift, um die Funktionen daraus zu nutzen.

#### 3.2.1.4 Others

##### Exceptions

In dem Namespace Common.Exceptions befinden sich selbst erstellte Exceptions die genutzt werden, um sie in der View abzufangen und darauf zu reagieren.

### 3.2.2 Verwendete Algorithmen, Bibliotheken oder Techniken

#### 3.2.2.1 NuGet Package MailKit

Für alles was mit E-Mails (abholen, senden, löschen, als gelesen markieren usw.) zu tun hat, wurde das MailKit verwendet. Es bietet einen ImapClient, Pop3Client und SmtpClient.

Der ImapClient und Pop3Client wurden für das Abholen der E-Mails verwendet und für die Testfunktion beim Account anlegen.

Der SmtpClient wurde nur zum Testen genommen, da er dafür einfacher einzusetzen war als der SmtpClient von .Net oder ein TcpClient. Für das Senden von E-Mails wurde der SmtpClient von .NET verwendet.

#### 3.2.2.2 WebBrowser

Für Anzeige von HTML-E-Mails wurde die WebBrowser View von C# genutzt.

### 3.2.3 Verwendeter fremder Quelltext

#### 3.2.3.1 Dispatcher

Von Herrn Rill habe ich zur Hilfe folgenden Code erhalten um eine E-Mail in einem Task zu den E-Mails eines Accounts hinzuzufügen:

```
var currentAccount = account;
var currentEmail = email;

dispatcher.BeginInvoke((Action)(() =>
{
    currentAccount.Emails.Add(currentEmail);
})));
```

Den Code habe ich auch noch an anderen Stellen verwendet, wo eine ähnliche Situation entstand.

#### 3.2.3.2 DataServices

Die Klasse DataServices habe ich von Herrn Rill seinem Beispiel Projekt „DataBindingAndMVVM“ aus der Solution „11 MVVMContactManager“ genommen und für mein Programm angepasst.

### 3.2.3.3 E-Mails abholen

Der Code um E-Mails abzuholen wurde von

[http://www.mimekit.net/docs/html/T\\_MailKit\\_Net\\_Imap\\_ImapClient.htm](http://www.mimekit.net/docs/html/T_MailKit_Net_Imap_ImapClient.htm) und

[http://www.mimekit.net/docs/html/T\\_MailKit\\_Net\\_Pop3\\_Pop3Client.htm](http://www.mimekit.net/docs/html/T_MailKit_Net_Pop3_Pop3Client.htm) verwendet. Er wurde aber kombiniert und erweitert.

Code:

```
using (var client = new ImapClient ()) {
    client.Connect ("imap.gmail.com", 993,
SecureSocketOptions.SslOnConnect);
    client.Authenticate ("username", "password");
    client.Inbox.Open (FolderAccess.ReadOnly);
    var uids = client.Inbox.Search (SearchQuery.All);
    foreach (var uid in uids) {
        var message = client.Inbox.GetMessage (uid);
        // write the message to a file
        message.WriteTo (string.Format ("{0}.msg", uid));
    }
    client.Disconnect (true);
}
```

### 3.2.3.4 E-Mails senden

Der Code um E-Mails zu senden wurde von <https://code-bude.net/2011/06/14/emails-versenden-in-csharp/> genommen und abgeändert.

Code:

```
public void sendMail(string absender, string empfaenger,
                    string betreff, string nachricht,
                    string server, int port,
                    string user, string passwort)
{
    MailMessage Email = new MailMessage();
    //Absender konfigurieren
    Email.From = new MailAddress(absender);
    //Empfänger konfigurieren
    Email.To.Add(empfaenger);
    //Betreff einrichten
    Email.Subject = betreff;
    //Hinzufügen der eigentlichen Nachricht
    Email.Body = nachricht;
    //Ausgangsserver initialisieren
    SmtpClient MailClient = new SmtpClient(server,port);
    if (user.Length > 0 && user != string.Empty)
    {
        //Login konfigurieren
        MailClient.Credentials = new System.Net.NetworkCredential(
passwort);
    }
    //Email absenden
    MailClient.Send(Email);
}
```

### 3.2.3.5 Kleinere Code-Fragmente

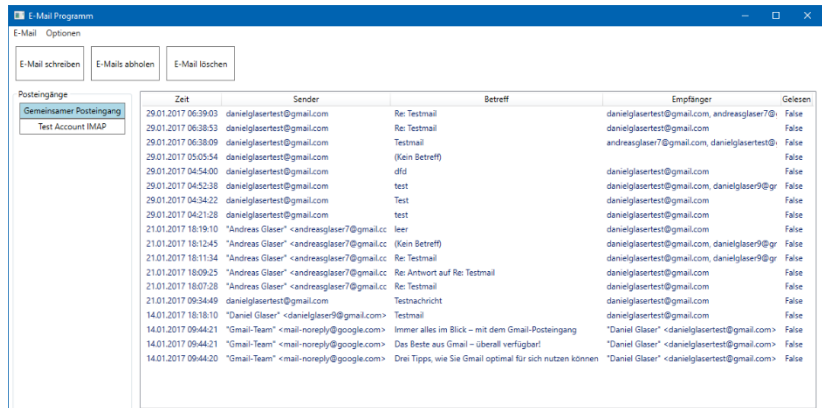
Kleinere Code-Fragmente wie z. B. das Sortieren einer ListView wurden im Code markiert. Aber zu kleine Code Fragmente (z. B. nur nachgeschaut, wie etwas heißt oder funktioniert) wurden nicht markiert.

## 4 Benutzerhandbuch

Bitte starten Sie als Erstes das Programm „E-Mail Programm“.

### 4.1 Hauptfenster

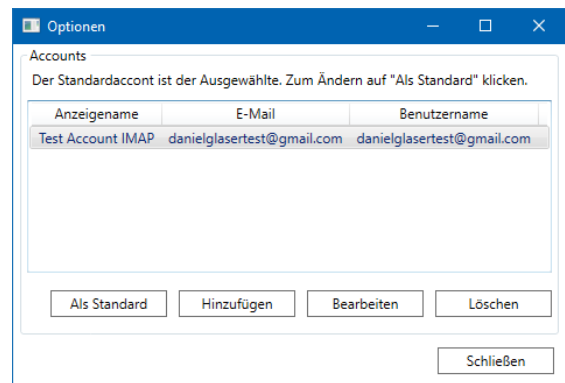
Im Hauptfenster können Sie Ihre E-Mail sehen. Links können Sie zwischen den Posteingängen Ihrer angelegten Accounts wechseln oder wie in der Standardansicht den gemeinsamen Posteingang sehen, in dem sich alle E-Mails aus all Ihren Accounts befinden.



### 4.2 Optionen

In den Optionen verwalten Sie Ihre Accounts.

Zum Öffnen von diesen gehen Sie bitte in der Menüleiste auf „Optionen“ und klicken auf „Optionen“. Dort können Sie einen Account anlegen, bearbeiten und löschen. Außerdem können Sie Ihren Standard Account für den E-Mail Versand wählen.



#### 4.2.1 Account anlegen

Klicken Sie bitte im „Optionen“-Fenster auf den Button „Hinzufügen“ um einen neuen Account anzulegen. Daraufhin öffnet sich ein neues Fenster, in dem Sie folgende Informationen eingeben müssen:

- Anzeigename
- Benutzername
- E-Mail
- Passwort
- IMAP/POP3-Server
- IMAP/POP3-Port
- IMAP oder POP3
- SMTP-Server
- SMTP-Port
- Signatur (optional)

Zum Testen, ob die eingegebenen Server und Ports richtig sind, können Sie die zwei Buttons „Test IMAP/POP3-Server“ und „Test SMTP-Server“ nutzen. Wenn diese grün werden, sind die eingegebenen Server und Ports richtig. Falls etwas nicht stimmen sollte, werden sie rot und es erscheint eine Fehlermeldung unter den Buttons, die Ihnen helfen soll, den Fehler zu finden.

Wenn Sie fertig sind, können Sie mit einem Klick auf den „Speichern“-Button den Account speichern.

#### 4.2.2 Account bearbeiten

Zum Bearbeiten eines Accounts gehen Sie bitte in die Optionen. Im „Optionen“-Fenster wählen Sie bitte den zu bearbeitenden Account aus und klicken auf den Button „Bearbeiten“. Alternativ können Sie auch auf den Account doppelt klicken. Daraufhin öffnet sich ein Fenster, in dem Sie den Account bearbeiten können. Die Vorgehensweise zum Bearbeiten ist genauso wie beim Account anlegen (4.2.1).



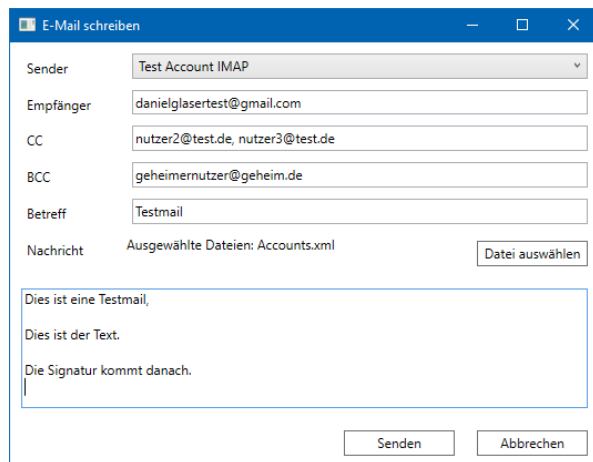
### 4.2.3 Account als Standard definieren

Um einen Account als Ihr Standard Account zum Versenden von E-Mails zu definieren, gehen Sie bitte in die Optionen und wählen den Account in der Liste aus den Sie als ihren Standard Account festlegen möchten und klicken Sie auf „Als Standard“.

### 4.2.4 Account löschen

Um einen Account zu löschen, gehen sie bitte in die Optionen und wählen den Account in der Liste aus den Sie löschen möchten und klicken Sie auf „Löschen“.

## 4.3 E-Mail schreiben



Um eine E-Mail zu schreiben, gehen Sie bitte in das Hauptfenster und klicken auf den Button „E-Mail schreiben“. Alternativ können Sie auch über die Menüleiste dorthin navigieren. Dazu in der Menüleiste den Punkt „E-Mail“ auswählen und dann „E-Mail schreiben“.

Es öffnet sich ein Fenster in dem Sie die Sendeadresse aus den Accounts, die Sie angelegt haben auswählen können. Im nächsten Feld können Sie die Empfangsadresse/n (bei mehreren mit „“ getrennt) eingeben an die die E-Mail gehen soll. Sie haben auch noch die Möglichkeit unter CC und BCC

E-Mail-Adressen einzugeben. Diese funktionieren wie standardmäßig vorgesehen.

In dem großen Eingabefeld können Sie nun Ihre Nachricht eingeben, die Sie schicken möchten. Sie können auch noch über den Button „Anhänge hinzufügen“ Dateien auswählen, die Sie mitschicken möchten.

## 4.4 E-Mails abholen

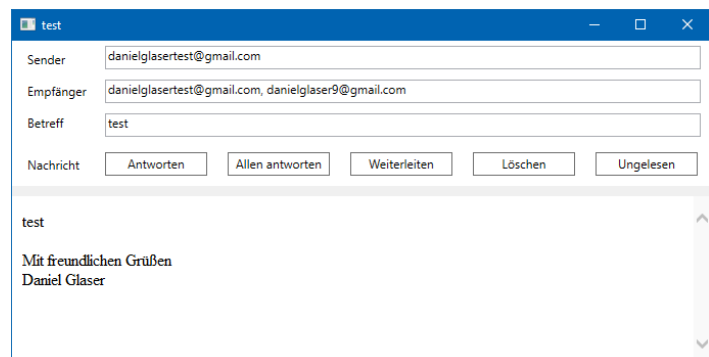
Bitte gehen Sie in das Hauptfenster der Anwendung und wählen Sie entweder einen Account aus oder den gemeinsamen Posteingang und klicken Sie auf „E-Mails abholen“. Daraufhin werden im Hintergrund die entsprechenden E-Mails abgeholt und im Hauptfenster aufgelistet.

## 4.5 E-Mails anzeigen

Bitte klicken Sie im Hauptfenster links auf den jeweiligen Posteingang, aus welchem Sie die E-Mails sehen möchten. In der Liste klicken Sie nun auf die E-Mail, die Sie sehen möchten, doppelt um sie zu öffnen.

### 4.5.1 E-Mail beantworten

Bitte öffnen Sie eine E-Mail und klicken Sie auf „Antworten“. Es öffnet sich das Fenster zum E-Mail schreiben. Schreiben Sie Ihre Nachricht und klicken Sie auf „Senden“. Wenn Sie allen antworten möchten, klicken Sie bitte auf „Allen antworten“.



### 4.5.2 E-Mail weiterleiten

Bitte öffnen Sie eine E-Mail und klicken Sie auf „Weiterleiten“. Es öffnet sich das Fenster zum E-Mail schreiben. Füllen Sie die Felder nach Belieben aus. Wenn Sie fertig sind, klicken Sie auf „Senden“.

### 4.5.3 E-Mail als ungelesen markieren

Bitte öffnen Sie eine E-Mail und klicken Sie auf „Ungelesen“.

## 5 Diskussion

---

### 5.1 Zusammenfassung der Ergebnisse

Die Grundfunktionen wie Senden und Empfangen von Text E-Mails funktionieren. Die Anzeige von HTML-E-Mails funktioniert. Das Antworten, allen antworten und weiterleiten von E-Mails funktioniert. Es ist möglich, mehrere Accounts mit IMAP oder POP3 anzulegen und zu bearbeiten. Zwischen den Posteingängen der Accounts kann gewechselt werden und alle E-Mails aller Accounts können unter Gemeinsamer Posteingang angesehen werden. E-Mails können lokal gelöscht werden und werden lokal auch als gelesen oder ungelesen markiert.

Was noch nicht funktioniert ist das Löschen von E-Mails auf dem Server. Das eine E-Mail gelesen wurde, wird bei IMAP noch nicht an den Server übermittelt. Weiteres siehe „Aufgetretene Probleme“ (5.2).

### 5.2 Aufgetretene Probleme

#### 5.2.1 WebBrowser ruckelt

Der WebBrowser ruckelt bei der Änderung der Größe des Fensters. Es konnte aber keine Lösung gefunden werden, liegt vermutlich daran, dass die WebBrowser View zu oft rendert.

#### 5.2.2 Testen des SMTP-Server

Das Testen des SMTP-Servers funktioniert noch nicht ganz so wie erwünscht. Näheres im Code (Klasse EmailService Zeile 133)

#### 5.2.3 Speichern von Anhängen

Das Speichern von Anhängen funktioniert noch nicht. Da ich von dem MailKit einen Anhang als MimeEntity bekomme der wiederum einen MimePart beinhaltet und ich diesen nicht serialisieren kann. Mögliche Lösung: Ich könnte die Anhänge gleich beim Abholen speichern und den Nutzer diese dann nur verschieben lassen.

#### 5.2.4 InvalidOperationException in der doEmailsContainsId

In der doEmailsContainsId tritt ab und zu eine InvalidOperationException auf. Auch obwohl ich nun eine neue List der ID erstelle, scheint sie immer noch ab und zu aufzutreten. Habe sie nun abgefangen und eine Ausgabe in der Konsole eingebaut, falls sie auftritt. Die Methode liefert dann, dass die E-Mail nicht enthalten ist und das Programm läuft einfach weiter.

#### 5.2.5 E-Mails abholen im gemeinsamen Posteingang

Die E-Mails tauchen erst auf, wenn man noch mal auf einen anderen Account und zurück wechselt. Das liegt daran, dass die E-Mails hier aus den ganzen Accounts gesammelt werden und beim Klicken auf den „Gemeinsamer Posteingang“-Button als ItemSource gesetzt werden.

Mögliche Lösung: Man müsste alle E-Mails in einer ObservableCollection halten oder eine eigene ObservableCollection für alle E-Mails anlegen. Ich habe zwar eine ObservableCollection für alle E-Mails erstellt aber nur in der Code-Behind, aber ohne das INotifyPropertyChanged Interface.

### 5.3 Weiterentwicklung des Programmes

Als Erstes sollten noch die Probleme behoben werden, die noch vorhanden sind. Danach könnte das Programm noch um weitere E-Mail Funktionen erweitert werden. Dazu zählen zum Beispiel:

- Anhänge wieder abwählen können
- Einen Namen eingeben können der zu der E-Mail-Adresse gehört
- Senden von HTML E-Mails
- E-Mail als Entwurf speichern

## 6 Eidesstattliche Erklärung

---

Ich versichere, dass ich die Studienarbeit ohne fremde Hilfe und ohne Benutzung anderer als der angegebenen Quellen angefertigt habe, und dass die Arbeit in gleicher oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen hat. Alle Ausführungen der Arbeit, die wörtlich oder sinngemäß übernommen wurden, sind als solche gekennzeichnet.

---

Ort, Datum

---

Unterschrift